Taking Routers Off Their Meds: Why Assumptions Of Router Stability Are Dangerous

Max Schuchard, Christopher Thompson, Nicholas Hopper, Yongdae Kim
University of Minnesota
{schuch, cthomp, hopper, kyd} @ cs.umn.edu

In this work, we examine how an adversary in control of a BGP speaker in a transit AS can cause a victim router in an *arbitrary* location on the Internet to become unstable. Through experimentation with both hardware and software routers, we look at the behavior of routers under *abnormal* conditions and come to four conclusions. First, routers placed in certain states behave in anything but a stable manner. Second, unexpected but *perfectly legal* BGP messages can place routers into those states with disconcerting ease. Third, an adversary can use these messages to disrupt a victim router to which he is not directly connected. Fourth, modern best practices do little to prevent these attacks.

Through experiments on hardware and software routers, we observed what happens when routers find themselves starved for CPU cycles or memory. We witnessed a variety of failure modes, ranging from severe performance degradation to the unrecoverable failure of all active routing sessions. We also observed that a router placed into one of these states would more than likely cause its peers to enter one or more of these states as well. An example of this is a CPU starved router was the exhausting its *peer's* memory. When the rate of incoming BGP updates exceeds a router's computational capacity, the receiving router only buffers a fixed number of incoming BGP messages. When those limits are reached it is up to the sender to buffer all future updates until the CPU starved router can accept them. We term this behavior *back pressure*. Figure 1 opposite shows the increase in memory usage over time for a router that is attempting to exchange routing tables with a CPU starved peer versus a peer with sufficient processing power.

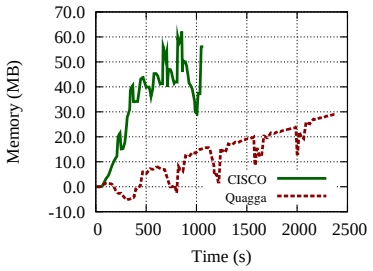

Figure 1.

We found it surprisingly easy to force a router into one of these unstable states. The majority of the methods we found are the result of taking commonly held assumptions about path attributes and invalidating those assumptions. Routers fail to handle these "corner cases" in a reasonable fashion. For example, in the software router Quagga uses a small, fixed size hash map with a predictable hashing function. This is acceptable so long as the assumption that AS paths will be spread evenly over all of the buckets holds. However, an adversary can violate this assumption, advertising AS paths that hash to the same value. Plots of the time to process updates with colliding AS paths compared to random AS paths can be seen in the figure opposite.

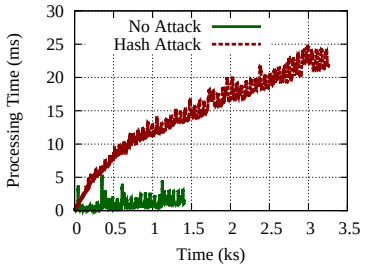

Figure 2.

An adversary in control of a BGP speaker can take advantage of these assumptions to attack other honest routers. By convincing legitimate BGP speakers to propagate these messages, an adversary can push a target in an arbitrary network location into unstable operation. A table below briefly highlights some current best practices, observations about how an adversary avoids them, and experimental evidence to support our observations. Details of how our adversary launches such an attack, along with how best practice fail to stop this can be found in our tech report [1].

Best Practice	Limitation	Experimental Evidence
Prefix Filters	Limits still allow millions of prefixes	/24s advertised by 88.5% of transit ASes
Prefix Aggregation	Not applied to routes from transit ASes	Hole punches and non-aggregated IP blocks
Prefix Limits	Malicious updates based on sum of victim prefix limits	Prefix limits applied on a per connection basis
AS Path Limits	Weakened by generous limits and memory allocation	Patsy allocates memory in fixed size blocks

[1] `http://www.cs.umn.edu/research/technical_reports.php`